**NumPy - Array Slicing**

Array slicing is a process of reading elements in a specific index range.

If the array is multi-dimensional, we can specify index in multi-dimensions to get the specific array slice.

**Slicing 1-D NumPy Array**

To slice a 1-D Array from specific starting position upto a specific ending position, use the following syntax.

arr[start:end]

Element at the end index is not included.

**1. Example to slice numpy array with index [1:5]**

In the following program, we take a 1-D NumPy Array, and do slicing with the slice [1:5].

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8])

print(arr[1:5])

Top of Form

Bottom of Form

**Output**

[2 3 4 5]

**Explanation**

array : [1, 2, 3, 4, 5, 6, 7, 8]

index : 0 1 2 3 4 5 6 7

| |

start end

slice : 2, 3, 4, 5

If start is not specified, then the starting index of the array 0 is taken. If end is not specified, then length of the array is taken.

**2. Example to slice 1D NumPy Array in steps**

We can also slice the array in steps.

The syntax to slice a 1-D NumPy Array in steps is

arr[start:end:step]

In the following program, we take a 1-D NumPy Array, and slice it in steps of 2.

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8])

print(arr[1:5:2])

Top of Form

Bottom of Form

**Output**

[2 4]

**Explanation**

array : [1, 2, 3, 4, 5, 6, 7, 8]

index : 0 1 2 3 4 5 6 7

| |

start end

step=2: \* \*

slice : 2, 4

**Slicing 2-D NumPy Array**

Just like a 1-D Array, we can also slice a 2-D Array.

To slice a 2-D Array from specific starting position upto a specific ending position, in the two dimensions, use the following syntax.

arr[start\_dim1:end\_dim1, start\_dim2:end\_dim2]

where

* start\_dim1:end\_dim1 is the start and end index of slice in the first dimension.
* start\_dim2:end\_dim2 is the start and end index of slice in the second dimension.

**1. Example to slice 2D numpy array with index [1, 1:3]**

In the following program, we take a 2-D NumPy Array, and slice it.

import numpy as np

arr = np.array([[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]])

print(arr[1, 1:3])

Top of Form

Bottom of Form

**Output**

[7 8]

**Explanation**

array : [[1, 2, 3, 4, 5], [6, 7, 8, 9, 10]]

1st dim (indexes) : 0 1

2nd dim (indexes) : 0 1 2 3 4 0 1 2 3 4

arr[1, :] : [6, 7, 8, 9, 10]

arr[1, 1:3] : 7, 8

Slicing [1, 1:3] means get the elements in the second row, from index 1 to 3 (element at index=3 not included).

**2. Example to slice 2D numpy array with index [0:2, 1:3]**

In the following program, we take a NumPy Array, and slice a 2x2 matrix .

import numpy as np

arr = np.array([[1, 2, 3, 4, 5],

[6, 7, 8, 9, 10],

[11, 12, 13, 14, 15],

[16, 17, 18, 19, 20]])

print(arr[0:2, 1:3])

Top of Form

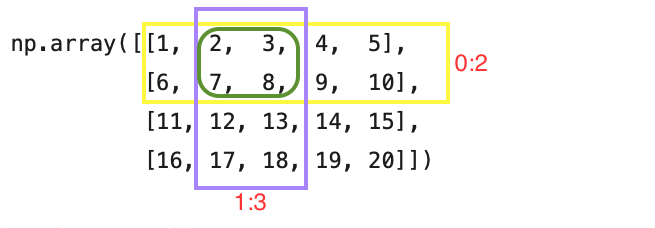
Bottom of Form

**Output**

[[2 3]

[7 8]]

**Explanation**



**Slicing 3-D NumPy Array**

As the number of dimensions increase, just increase the number of slices in the square brackets, separated by comma.

To slice a 3-D Array from specific starting position upto a specific ending position, in different dimensions, use the following syntax.

arr[start\_dim1:end\_dim1, start\_dim2:end\_dim2, start\_dim3:end\_dim3]

where

* start\_dim1:end\_dim1 is the start and end index of slice in the first dimension.
* start\_dim2:end\_dim2 is the start and end index of slice in the second dimension.
* start\_dim3:end\_dim3 is the start and end index of slice in the third dimension.

Instead of both start:end, we can specify only a single index. In that case, element at that index is only selected for slicing.

In the following program, we take a 3-D NumPy Array, and slice it.

import numpy as np

arr = np.array([[[1, 2, 3, 4, 5],

[6, 7, 8, 9, 10]],

[[11, 12, 13, 14, 15],

[16, 17, 18, 19, 20]]])

print(arr[0, 1, 1:4])

Top of Form

Bottom of Form

**Output**

[7 8 9]

**NumPy - Split array into smaller arrays**

To split a numpy array into smaller arrays, you can use the numpy function numpy.array\_split().

The syntax to call array\_split() function is

numpy.array\_split(arr, n)

where

* arr is the array you want to split.
* n is the number of smaller arrays you want to split array arr into.

We can also use numpy.split() function. The syntax of split() function is

numpy.split(arr, indices, axis)

where

* arr is the array you want to split.
* indices are the indices at which you want to split the array arr.
* axis is the axis along which the split has to be done.

**Examples**

**1. Split array into 4 smaller arrays**

In the following program, we take a numpy array (one dimensional) of length 12, and split the array into 4 smaller arrays.

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])

output = np.array\_split(arr, 4)

print(output)

Top of Form

Bottom of Form

Copy

**Output**

[array([1, 2, 3]), array([4, 5, 6]), array([7, 8, 9]), array([10, 11, 12])]

**2. Split array at specific indices**

In the following program, we take a numpy array (one dimensional), and split the array at indices 3 and 7.

import numpy as np

arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])

output = np.array\_split(arr, [3, 7])

print(output)

Top of Form

Bottom of Form

**Output**

[array([1, 2, 3]), array([4, 5, 6, 7]), array([ 8, 9, 10, 11, 12])]

**Explanation**

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12] <- arr

0 1 2 3 4 5 6 7 8 9 10 11 <- indices

|3 |7

[1, 2, 3][4, 5, 6, 7][8, 9, 10, 11, 12] <- splits